11/22/2019

[林晨]

[17302010021]

[Lab 2 设计/功能文档]

[Check Point]

1. RPC协议的选择

选择现有的java RMI框架实现远程过程调用。目前猜想有实现转接口adapter的方式如下：

参考课程群中同学给的转接口代码样例和网上RMI的解析。首先创建一个中间接口Adapter用于服务器端，然后对需要传输的实体创建类，在此基础上实现该接口。客户端包含该中间接口为属性，由此来调用实现适配器的对象的方法。

实际上，转接口实现服务器端对于FileManager和BlockManager的转接。这个过程中通过一个实现转接口的类来包装他们，从而使得对象可以绑定至端口。

1. 描述流程与交互

**读写流程与交互过程**：读取时客户端输入读取指令和读取文件名和内容长度。与NFS不同，服务器端维护文件状态（游标等）。服务器端获得指令和信息，然后读取相应的内容后包装发送给客户端。此时本人要注意文档中提到的序列化要求。然后客户端即可根据获得的信息进行打印等操作。写入时有两种形式：一种要创建新的文件。那么此时首先要发送给服务器申请创建文件。然后获得服务器创建成功消息后由服务器发送相关信息。接着进入第二种形式：客户端将要写入信息包装发送，服务器收到后调用对应FileManager进行处理写入，最后确认成功。

**涉及的对象**：目前涉及BlockManagerImpl、FileManagerImpl、还有服务器端实现了转接口的RawBlockManager和RawFileManager以及原本的BlockImpl和FileImpl。除此之外，BlockManagerServer和FileManagerServer分管文件管理器服务和块管理器服务。一种方式：其中FileManagerServer应当间接控制BlockManagerServer的启用（FileManagerServer所涉及的Block所在的BlockManager）。另一种方式：FileManagerServer与BlockManagerServer完全分离，即FileManager的启用与BlockManager的启用没有关联。从而读取时候可能会由于某些BlockManager未启用而抛出异常。（视最终情况决定采取什么方式）

**调用的接口**：转换器接口BMAdapter和FMAdapter。由于FileManager和BlockManager都没有转化为继承Remote的形式，转接口为此提供不改变原本接口上的转接作用。另外，还有Lab1中设计的接口调用。

**程序状态变化**：两个分离的Server端分别自启动。启动后可选择开启或者关闭相应的FileManager或者BlockManager服务。

客户端承担发送请求和处理部分可处理异常的任务。

启动状态下服务器主要有启动服务、等待请求、处理请求、返回消息等几种状态。客户端有发送消息、等待消息、接收消息、处理接受到的消息等状态。

1. 简述

①Server端的 Manager 是如何启动的？

FileManagerServer和BlockManagerServer都各自启动。即依赖自身的静态方法入口。在Server启动后，该入口会循环询问是否要进行唤醒某个FileManager或者某个BlockManager，或者睡眠某个FileManager或者某个BlockManager。服务器端人员自行输入决定唤醒（启动）或者睡眠Manager，由此模拟实现了Manager在Server端任何时刻的启动。

②Server端的 Manager 在异常终⽌时如何做到尝试重启的？

在本Lab中，由于我的启动采用了Server端人员输入的形式。那么对于异常终止情况，可以直接采用重新调用程序入口的方式重启整个过程，Server端人员可以重新输入以开启或者关闭某个Manager。

③Client端是如何实现超时检查这个特性的？

客户端的检查超时预计会重写createSocket方法，其中加入socket.connect（new InetSocketAddress（host，port），500）的设置，将时长限制在500ms。也可以采用网上查到的java -Dsum.rmi.transport.tcp.responseTimeout来设置。

④Manager异常下线

1. 已经上线的FileManager此时其中的一个文件对应的几个BlockManager下线了。解决：此时读取部分内容将不能得到。此时会向客户端发出异常，并提醒用户之后等待Manager上线再试。
2. 一个文件相关的所有FileManager和BlockManager均在线。但是突然FileManager下线了。解决：那么此时对于后面的读取和写入操作一律禁止，返回客户端用户异常。
3. Manager异常终止程序。解决：此时需要在过程中记录日志，从而在下一次启动时回复原本的状态。